ESPDA: ENERGY-EFFICIENT AND SECURE PATTERN-BASED DATA AGGREGATION FOR WIRELESS SENSOR NETWORKS

H. Çam, S. Özdemir, P. Nair[*]*, D. Muthuavinashiappan*

Department of Computer Science and Engineering,
[*]Department of Electrical Engineering
Arizona State University,
Tempe, AZ 85287

ABSTRACT

Secure data transmission and data aggregation are critical in designing cluster-based sensor networks. This paper presents an Energy-efficient and Secure Pattern-based Data Aggregation protocol (ESPDA) for wireless sensor networks. ESPDA is energy and bandwidth efficient because cluster-heads prevent the transmission of redundant data from sensor nodes. ESPDA is also secure because it does not require the encrypted data to be decrypted by cluster-heads to perform data aggregation. In ESPDA, cluster-head first requests sensor nodes to send the corresponding pattern code for the sensed data. If multiple sensor nodes send the same pattern code to the cluster-head, then only one of them is permitted to send the data to the cluster-head. Hence, ESPDA has advantages over the conventional data aggregation techniques with respect to energy, bandwidth efficiency and security. Simulations results show that as data redundancy increases, the amount of data transmitted from sensor nodes to cluster-head decreases up to 45% when compared to conventional algorithms.

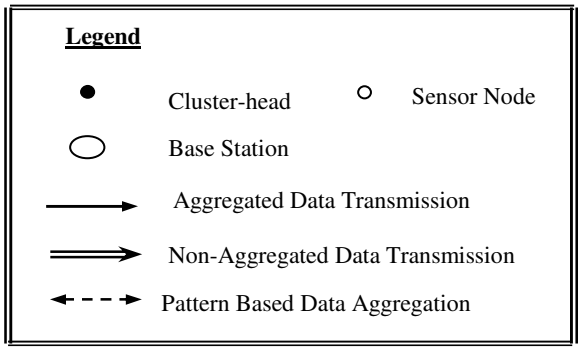
1. INTRODUCTION

Wireless sensor networks are severely energy-constrained systems in which each sensor node has a limited power and computation capability [1] [2]. They are essentially event-based systems with a broad range of environmental sensing applications from vehicle tracking to habitat monitoring. A sensor network usually consists of a large number of sensor nodes divided into clusters. Base station collects and processes the data from clusters. Depending on the architecture, there might be more than one level of cluster-heads between sensor nodes and the base station.

This paper proposes an Energy-efficient and Secure Pattern-based Data Aggregation protocol (ESPDA) for cluster-based wireless sensor networks. Knowing that 70% of the energy consumption is due to data transmission [3], the proposed ESPDA reduces data transmission by not sending the redundant data from sensor nodes to cluster-heads. Since the number of sensors in a sensor network is very large, often various sensors detect common data. Data aggregation [4] is used to eliminate redundancy and minimize the number of transmissions in order to save energy. In conventional data aggregation methods, cluster-heads receive all the data from sensor nodes and then eliminate the redundancy by checking the contents of the data as shown in Figure 1(a). In ESPDA, instead of transmitting the entire data with redundancy, the sensor nodes send the corresponding pattern codes to cluster-head for data aggregation. Thus, data aggregation is performed even before the actual data is transmitted from the sensor nodes as illustrated in Figure 1(b).

Security in data communication is another important issue to be considered while designing wireless sensor networks. Since ESPDA aggregates data by pattern codes, cluster-heads need not know the contents of the transmitted data, which enables ESPDA to work in conjunction with our security protocol [5] where sensor data is transmitted to base station in encrypted form without decrypted anywhere in the transmission path. Moreover, pattern codes are generated using a secret pattern seed which prevents from retrieving the real data from pattern codes.

Although data aggregation and security in wireless sensor networks has been studied extensively, to the best of our knowledge, there is no previous research considering data aggregation and security together.



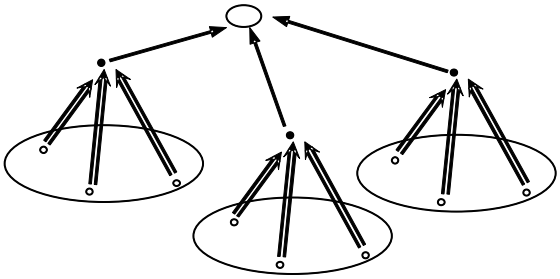


Figure 1(a). Data Transmission using conventional data aggregation.

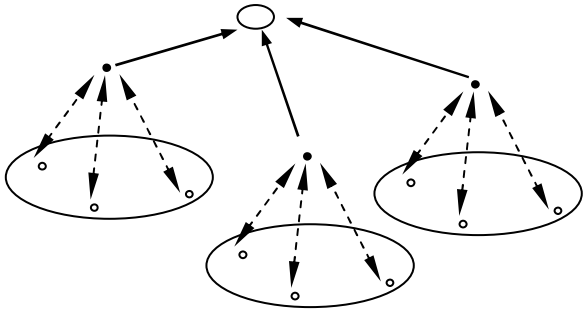


Figure 1(b). Data Transmission using ESPDA technique

2. ESPDA PROTOCOL

Considering the energy constraints in wireless sensor networks the proposed ESPDA is designed to provide energy-efficient data aggregation together with secure data communication. ESPDA protocol consists of a number of algorithms and processes.

Sensor nodes implement the following processes.

- Sensing data from the environment.
- Defining intervals from threshold values set for the environment parameters.
- Assigning critical values for intervals using pattern seed from cluster-head.
- Generating the lookup table.
- Generating pattern codes using pattern generation algorithm.
- Sending pattern codes to cluster-heads.
- Receiving send-requests/ACK from the cluster-head
- Sending actual data to cluster-heads.

Cluster-head performs the following processes.

- Broadcasting the pattern seed for each time interval.
- Receiving pattern codes from sensor nodes.
- Forming the selected-set of pattern codes using the pattern comparison algorithm.
- Requesting selected sensor nodes to send actual data.

The pattern generation and comparison algorithms as well as an example for pattern generation are presented in the sequel.

2.1 Pattern Generation

Sensor nodes receive the secret pattern seed from the cluster-head. The interval values for the data are defined, based on the given threshold values set for each environment parameter. The number of threshold values and the variation of intervals may depend on the user requirement and the precision defined for the given environment in which the network is deployed. The algorithm then computes the critical values for each interval using the pattern seed to generate the lookup table, where the pattern seed is a random number generated and broadcasted by the cluster-head.

This pattern seed is changed at regular time intervals. In ESPDA, the pattern generation algorithm (PG) which is executed on all sensor nodes uses the pattern seed to generate pattern codes. Before sending the actual data, the sensor nodes send the pattern codes to the cluster-head. These patterns are analyzed by the pattern comparison algorithm at the cluster-head to prevent redundant data being transmitted Sensor nodes sends the set of unique data (without redundancy) to the cluster-head which is transmitted to the base station.

Critical values form the base for the generation of pattern codes. Now, when data is sensed from the environment, its characteristics are compared with the intervals defined in the lookup table of PG algorithm and a corresponding critical value is assigned. The pattern code is generated by combining the critical values of all parameters of the data.

ALGORITHM: **Pattern Generation (PG)**

Input: Environment parameters, type of sensed data, threshold levels (interval) and precision of data

Output: Pattern-codes (**PC**)

Begin

1. Declare arrays for interval [100], lookup[100][100]
2. Variables PC = ‘’, seed = null
3. **if** (new seed is received) **then**
4. **for** i = 1 to n

5. interval [i] = threshold[i-1] '-' threshold [i]
6. **endfor**
7. Assign a critical value for each interval created (for e.g. min-value '1' for the first interval to max-value '9'for the last interval. This assignment of values varies when the seed changes).
8. Form the critical value lookup table by updating all the critical values and corresponding intervals for each type of data sensed.
9. **endif**
10. **while** (data sensed 'D' is available)
11. Get the actual data sensed from the environment.
12. Round off data for require precision.
13. Find the respective critical value for each current data sensed using lookup table.
14. PC = PC + critical value (Append value)
15. Repeat step 13, 14 & 15 until all parameters of the data for that timestamp are done.
16. Send final PC, timestamp, sensor ID to cluster-head.
17. **endwhile**

End

The lookup table is re-generated whenever a new pattern seed is broadcasted by the cluster-head. Since the pattern seed is periodically changed to prevent the intruders from manipulating the data by listening to the pattern codes for a long time, this technique enforces security as well as data freshness.

2.2 Pattern Comparison

The cluster-head also has equal responsibility as the sensor nodes in data aggregation. It sends the pattern seed periodically to all active sensor nodes to maintain the confidentiality of the pattern codes. After receiving pattern codes from the sensor nodes for a time period T, the entire set of codes is classified based on redundancy. Unique patterns are then moved to the 'selected-set' of codes. The time period T varies based on the environment where the sensor network is deployed. The sensors nodes that correspond to the unique pattern set ('selected-set') are then requested to transmit the actual data. ACK signals may be broadcasted to other sensors ('de-selected-set') to discard their (redundant) data. These sensor nodes can be put to sleep mode to conserve power.

ALGORITHM: **Pattern Comparison**

Input: Pattern codes

Output: Request sensor nodes in the selected-set to send actual encrypted data.

Begin

1. Broadcast 'current-seed' to all sensor nodes
2. **while** (current-seed is not expired)
3. time-counter = 0
4. **while** (time-counter < T)
5. **get** pattern code, sensor ID, timestamp
6. **endwhile**
7. Compare and classify pattern codes based on redundancy to form 'classified-set'.
8. selected-set={one pattern code from each classified-set}
9. deselected-set = classified-set – selected-set
10. **if** (sensor node is in selected-set)
11. Request sensor node to send actual data
12. **endif**
13. **endwhile**

End

This technique ensures that the sensed data cannot be re-generated from the pattern codes which in turn help the sensor nodes to send pattern codes to cluster-head without any encryption. Besides, the pattern seed is known only to sensor nodes in the cluster, therefore the pattern codes ensure the security of the sensed data during the data aggregation. The security of actual data transmission is provided by our security protocol [5].

2.3 Example

Considering there are 5 sensor nodes sensing temperature (d1), pressure (d2) and humidity (d3) in a given environment. Each parameter sensed is assumed to have threshold values between the ranges 0 to 100 as shown in Table 1.

Threshold values	30	50	70	80	90	95	100
Interval values	0-30	31-50	51-70	71-80	81-90	91-95	96-100
Critical values	5	3	7	8	1	4	6

Table 1. Look up table for critical values.

At any instant of time the sensor node senses the data 'D' (d1, d2, d3) from the environment. Algorithm PG computes the pattern codes as shown in Table 1 by assigning critical values to the data using Table 1. Data sensed by sensor 1 and sensor 3 is redundant. Similarly data sensed by sensor 2, sensor 4 and sensor 5 are redundant. Hence the cluster-head

selects only sensor 1 and sensor 4 to transmit the data from each redundant set, based on the timestamps.

	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5
Data	D(56, 92, 70)	D(70, 25, 25)	D(58, 93, 69)	D(68, 28, 30)	D(63, 24, 26)
Critical value for d1	7	7	7	7	7
Critical value for d2	4	5	4	5	5
Critical value for d3	7	5	7	5	5
Pattern code	747	755	747	755	755

Table 2. Pattern codes generation table.

3. SECURE DATA AGGREGATION

Wireless sensor networks are severely energy constrained and hence every operation of the nodes including the data communication should be made energy efficient. Asymmetric cryptographic algorithms are not suitable to provide security on wireless sensor networks since they require high computation power, and storage resources. Therefore, symmetric key cryptographic algorithms are employed to support security in our wireless sensor networks [5]. Nevertheless, these algorithms also compromise security because of limited key length and memory available on the sensor nodes. In order to mitigate this shortcoming of symmetric cryptographic algorithms we employ Non-blocking Orthogonal Variable Spreading Factor (NOVSF) [7] code hopping technique in addition to changing session keys dynamically.

In order to perform data aggregation, generally the data transmitted by the sensor nodes should be decrypted at the cluster-head. The aggregated data is then encrypted before being transmitted to the base station. This technique is vulnerable from security perspective because decryption of data requires the cluster-head to obtain the symmetric key. In ESPDA, since cluster-head does not decrypt the data the protocol is more secure. By implementing ESPDA we eliminate this intermediate process, which reduces the overhead of the cluster-heads and thus contributing to energy efficiency. The data aggregation is done before the actual data is transmitted by the sensor nodes. In what follows we give a brief description of how our security protocol [5] works in conjunction with ESPDA. The sensor nodes have a unique secret built in key. The base station, periodically broadcasts a session key (different from pattern seed used in ESPDA) to maintain data freshness. The sensor node computes a node-specific-secret-key (NSSK) using the session key and the built in key. This NSSK is used to encrypt and decrypt all the consequent data transmission during that session. The base station has the knowledge about all the unique built in keys of the sensor nodes, which is used to compute NSSK at the base station for decryption. The detailed explanation of the security protocol as well as NOVSF code-hopping technique is given in [4].

4. PERFORMANCE ANALYSIS

In this section, we first compare the energy efficiency of conventional data aggregation algorithm with ESPDA. In conventional data aggregation, unlike ESPDA, cluster-head eliminates the redundancy after obtaining the entire actual data from sensor nodes. In what follows we will show that ESPDA is more energy efficient than the conventional data aggregation technique because the number of transmitted packets in ESPDA is much less than the conventional one.

Let us consider T as the total number of packets that sensor nodes want to transmit in a session, and R as the number of distinct packets, where R less than or equal to T.

In conventional data aggregation algorithms since the cluster-head receives all data packets prior to eliminating redundant data, the total number of packets transmitted from sensor nodes to cluster-head would be T. After eliminating redundancy the cluster-head sends R packets to base station. Therefore, the total number of packets transmitted from sensor nodes to base station is $(T+R)$.

In ESPDA cluster-head receives T pattern codes from all sensor nodes. After eliminating redundancy based on pattern codes, cluster requests selected sensor nodes to transmit their data. Since selected nodes are the nodes that have distinct packets, the total number of packets transmitted from sensor nodes to cluster-head would be R which are later transmitted to base station. Therefore, the total number of packets transmitted from sensor nodes to base station is $(2R)$.

In wireless sensor networks, often various sensor nodes detect common data and hence R is usually much less than T. Therefore, ESPDA is energy efficient when compared to the conventional data aggregation algorithm.

To assess the energy efficiency of ESPDA, we wrote a simulator to simulate the ESPDA protocol. GloMoSim [6] is used to simulate the transmission of data and pattern codes from sensor nodes to cluster-head. Simulation results show that ESPDA improves energy efficiency significantly by reducing the number of packets transmitted in data communication as shown in Figure 2. The pattern code generation requires negligible amount of energy as the algorithm is not complex. The energy required for transmission of pattern codes in ESPDA is also negligible since pattern codes consist of few bits.

In our simulations we have considered the communication channel bandwidth between the sensor nodes and base station. The occupied bandwidth rate is the ratio of bandwidth occupancy and the total available bandwidth. When compared to conventional data aggregation algorithms, as the redundancy increases the bandwidth efficiency of ESPDA also increases (Figure 2.). At 100% redundancy, the bandwidth occupancy of ESPDA is close to zero, since ESPDA eliminates redundancy before sensor nodes transmit the actual data packets. However, in conventional data aggregation bandwidth occupancy is more than 50% of the total bandwidth since all sensor nodes transmits the actual data to be aggregated at cluster-head.

Since, ESPDA works in conjunction with our security protocol, the performance of our security system also affects ESPDA. The performance analysis of our security protocol with previous security systems shows that our protocol does not increase the data payload [5]. The memory and computational energy required by our security protocol is comparatively less, since the protocol uses short symmetric keys.

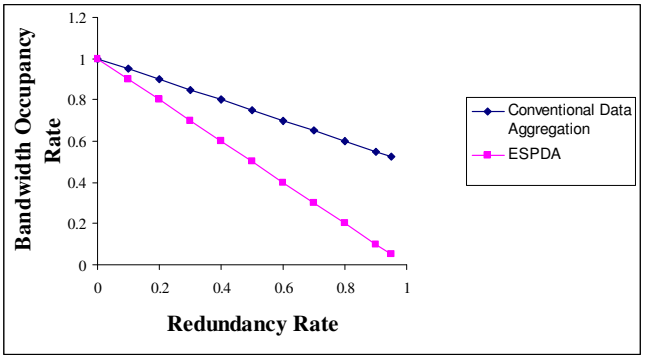


Figure 2: Occupied bandwidth rate versus redundancy rate for ESPDA and conventional data aggregation.

5. CONCLUSION

This paper has introduced an energy-efficient and secured data aggregation protocol called ESPDA. In contrast to conventional data aggregation protocols ESPDA avoids the transmission of redundant data from the sensor nodes to the cluster-head. To make the data transmission and aggregation more secured cluster-head is not required to decrypt or encrypt the data received from the sensor nodes. The symmetric keys that are used due to their low memory space and computing requirements, are not transmitted between the cluster-head and the sensor nodes. Simulation results show that ESPDA improves the energy and bandwidth efficiency the protocol reduces the number of packets transmitted. Thus when ESPDA is integrated with our previously proposed security protocol [5] it greatly helps to achieve the primary goal of energy efficiency and security essential in wireless sensor networks.

Future work includes the study of the amount of redundant data for some applications such as temperature control. More research will be done with bigger sensor network and considering network complexity in pattern code generation algorithm.

6. REFERENCES

[1] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", *Proc. of INFOCOM 2002*, vol. 3, pp. 1567-1576, June 2002.

[2] A. Sinha and A. Chandrakasan, "Dynamic power management in wireless sensor networks", *IEEE Design and Test of Computers*, vol. 18(2), pp. 62-74, March-April 2001.

[3] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, and D.E. Culler, "SPINS: Security protocols for sensor network", *Wireless Networks*, vol. 8, no. 5, pp. 521-534, 2002.

[4] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on Data Aggregation in wireless sensor networks", *Proc. of the 22nd International Conference on Distributed Computing Systems*, pp. 575-578, July 2002.

[5] H. Çam, S. Özdemir, D. Muthuavinashiappan, and Prashant Nair, "Energy-Efficient security protocol for Wireless Sensor Networks", *IEEE VTC Fall 2003 Conference*, October 2003, Orlando, Florida.

[6] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks", *Proc. of the 12th Workshop on Parallel and Distributed Simulations*, PADS'98, May 1998, Banff, Alberta, Canada.

[7] H. Çam and K. Vadde, "Performance analysis of Non-blocking OVSF codes in WCDMA", in Proc. of *The 2002 International Conference on Wireless Networks*, pp. 50-55, June 2002.